

# System Automation mit Puppet und Foreman

Toni Schmidbauer

26. März 2014

# whoami

- ▶ SysAdmin@s-itsolutions
- ▶ toni@stderr.at
- ▶ <http://github.com/tosmi>
- ▶ stderr@jabber.org

# Agenda

- ▶ Kurze Umfrage
- ▶ Was ist Puppet?
- ▶ Was ist Foreman?
- ▶ Puppet@s-iTSolutions
- ▶ Was haben wir geplant

# Umfrage

# Was ist Puppet?

```
1  class linuxwochen2014 {  
2  
3      user { 'linuxwochen':  
4          ensure => present,  
5          uid => 4711,  
6          gid => 4711,  
7      }  
8  
9      package { 'emacs':  
10         ensure => installed  
11     } ->  
12     package { 'vi':  
13         ensure => absent,  
14     }  
15 }
```

# Zuordnung von Klassen

```
1 node linuxwochen {  
2     include mypuppetconfig  
3 }
```

- ▶ oder über einen External Node Classifier (Foreman)

# Was ist Foreman?

 FOREMAN

Dashboard Hosts Reports Facts Audits Statistics Trends More ▾

## Overview

Filter ...   ▾

### Host Configuration Status

Hosts that had performed modifications without error	3
Hosts in error state	11
Good host reports in the last 35 minutes	875
Hosts that had pending changes	0
Out of sync Hosts	67
Hosts with no reports	4
Hosts with alerts disabled	5

Total Hosts: 965

OK 91%

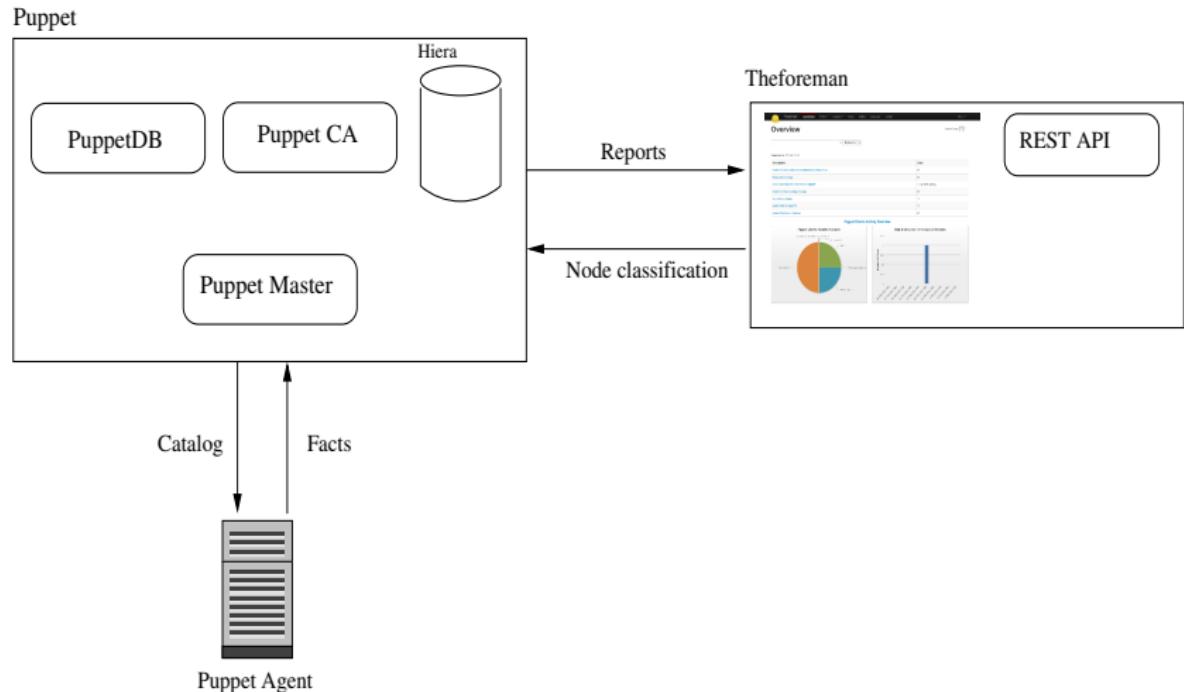
### Latest Events

Host	A	R	F	FR	S	P
appplusu2.l...	0	0	0	0	8	0
appqsys2.se...	0	0	0	0	8	0
appplusu2.i...	0	0	0	0	8	0
appqsys2.se...	0	0	0	0	8	0
appplusu2.l...	0	0	0	0	8	0
appqsys2.se...	0	0	0	0	8	0

### Run distribution in the last 30 minutes

Time Interval	Runs
1	85
2	80
3	95
4	85
5	90
6	92
7	80
8	90
9	95
10	85
11	90
12	75

# Puppet run



# Und jetzt?



- ▶ Wie verwaltet wir unseren Puppet Code?
- ▶ Wie soll unsere Puppet Umgebung aussehen?
- ▶ Wie erfolgt das Deployment des Codes?
- ▶ Wie organisieren wir Module?
- ▶ Wie soll eine Entwicklungsumgebung aussehen?
- ▶ Wie testen wir den Puppet Code?
- ▶ Wie verwalten wir Module von PuppetForge?

Wie verwaltet wir unseren Puppet  
Code?

Wie organisieren wir Module?

# GIT

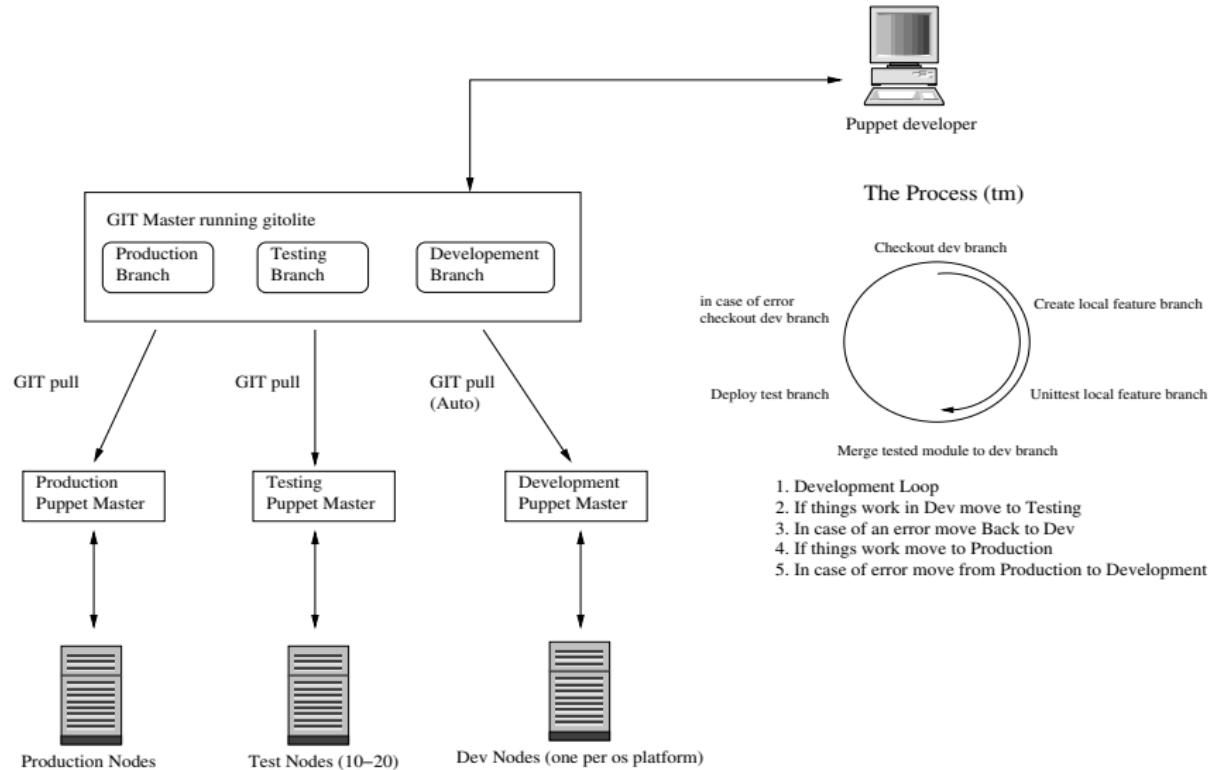
- ▶ Ein zentrales GIT Repository
- ▶ Berechtigungssystem für GIT mit gitolite
- ▶ 3 Hauptbranches
  - ▶ Master: Staging via GIT pull auf 4 Dev Server
  - ▶ Testing: ca. 25 "Produktions" Server (git pull)
  - ▶ Production: der Rest, Staging via tags
- ▶ Feature Branches für neue Module

# Demo

Wie soll unsere Puppet Umgebung aussehen?

Wie erfolgt das Deployment des Codes?

# Puppet Umgebung



# Puppet Umgebung II

- ▶ Am Development Master verwenden wir dynamische Puppet Environments

<http://puppetlabs.com/blog/git-workflow-and-puppet-environments>

- ▶ Jeder Feature Branch wird ein eigenes Puppet Environment
- ▶ Das Staging in die Produktion erfolgt über GIT tags

# Wie soll eine Entwicklungsumgebung aussehen?



# Vagrant

- ▶ <http://vagrantup.com>
- ▶ Ermöglicht virtuelle Entwicklungsumgebungen
- ▶ Vagrant Box ist ein vorkonfiguriertes Image
- ▶ Default VirtualBox andere Provider via Plugins (VMWare, KVM)

# Demo

# Wie testen wir den Puppet Code?

# rspec-puppet

- ▶ Ruby RSpec Tests für Puppet
- ▶ Ist in unserer Vagrant Umgebung vorinstalliert
- ▶ Jedes Module muss RSpec Tests mitbringen
- ▶ Commit in Master Branch löst einen Jenkins Build aus
- ▶ Commit in Testing Branch löst einen Jenkins Build aus
- ▶ Commit in Production Branch löst einen Jenkins Build aus

# Wie verwalten wir Module von PuppetForge?

# Puppetforge Module

- ▶ Eigenes GIT Repository (puppetforge.git)
- ▶ Download der Module in der Entwicklungsumgebung
- ▶ Staging wie unser Haupt Puppet Repository

- ▶ Wie verwaltet wir unseren Puppet Code? *DONE*
- ▶ Wie soll unsere Puppet Umgebung aussehen? *DONE*
- ▶ Wie erfolgt das Deployment des Codes? *DONE*
- ▶ Wie organisieren wir Module? *DONE*
- ▶ Wie soll eine Entwicklungsumgebung aussehen? *DONE*
- ▶ Wie testen wir den Puppet Code? *DONE*
- ▶ Wie verwalten wir Module von PuppetForge? *DONE*



# Probleme, Probleme, Probleme...

- ▶ Ein GIT Repo funktioniert nicht bei Änderungen von Upstream Modulen
- ▶ Andere Abteilungen sollen ihre Module unabhängig Testen
- ▶ Unitests sagen noch nichts aus wie sich der Code am Live-System verhält
- ▶ Wir sollten eigentlich das Zusammenspiel aller Modelle testen (Forge und eigene)

# Was haben wir geplant?

- ▶ r10k für Deployment
- ▶ Ein Repository pro Module
- ▶ Puppet coverage tests
- ▶ Acceptance Tests mit Beaker